

CLAIMS

What is claimed is:

- 1 1. A computer system, comprising:
 - 2 a pipelined, simultaneous and redundantly threaded ("SRT") processor comprising a fetch
 - 3 unit that further comprises a branch predictor;
 - 4 an I/O controller coupled to said processor;
 - 5 an I/O device coupled to said I/O controller; and
 - 6 a main system memory coupled to said processor;
 - 7 wherein said SRT processor processes a set of instructions in a leading thread and also in a
 - 8 trailing thread and the SRT processor speculates on the outcome of branch instructions in the
 - 9 leading thread using the branch predictor, but wherein the SRT processor does not speculate on the
 - 10 outcome of branch instructions in the trailing thread and instead uses the actual outcome of branch
 - 11 instructions in the leading thread to predict the outcome of branch instructions in the trailing
 - 12 thread.
- 1 2. The computer system of claim 1 further comprising a branch outcome queue located in the
- 2 fetch unit;
- 3 wherein the actual outcomes of branch instructions in the leading thread are placed in the
- 4 branch outcome queue.
- 1 3. The computer system of claim 2 wherein the fetch unit accesses the branch outcome queue
- 2 and not the branch predictor to predict the outcome of branch instructions in the trailing thread.

1 4. The computer system of claim 2 wherein the branch instruction queue is a FIFO buffer.

1 5. The computer system of claim 2 wherein the individual branch outcome entries in the
2 branch outcome queue comprise a program type identifier and a target address for the location of
3 the next instruction in the thread to be executed.

1 6. The computer system of claim 2 further comprising a register update unit;
2 wherein the register update unit is configured to hold instructions in a queue until the
3 instructions are executed and retired by the SRT processor and wherein the outcomes of branch
4 instructions in the leading thread are not placed in the branch outcome queue until the branch
5 instructions retire from the register update unit.

1 7. The computer system of claim 2 further comprising a slack counter located in the fetch
2 unit;
3 wherein the slack counter is configured to maintain an approximately constant number of
4 instructions of separation between corresponding instructions in the leading and trailing threads.

1 8. The computer system of claim 3 wherein if the branch outcome queue becomes full,
2 execution of instructions in the first thread is temporary halted to prevent more branch outcomes
3 from entering the branch outcome queue; and
4 wherein if the branch outcome queue becomes empty, execution of instructions in the
5 second thread is temporary halted to allow more branch outcomes to enter the branch outcome
6 queue.

2 a slack counter configured to maintain a target number of instructions of separation
3 between corresponding instructions in the leading and trailing threads.

1 12. The SRT processor of claim 9 wherein said fetch unit comprises:

2 a branch predictor for predicting the outcomes of branch instructions in the first program
3 thread; and

4 a branch outcome queue for storing the actual outcomes of branch instructions in the first
5 program thread;

6 wherein the actual outcomes of branch instructions in the first program thread are stored in
7 the branch outcome queue after the branch instructions in the first program thread are retired by the
8 SRT processor; and

9 wherein the fetch unit uses the branch outcome queue and not the branch predictor to
10 predict the outcomes of branch instructions in the second program thread.

1 13. The SRT processor of claim 12 wherein the SRT processor is an out-of-order processor
2 capable of executing instructions in the most efficient order, but wherein branch instructions are
3 executed in the same order in both the first and second program threads.

1 14. The SRT processor of claim 13 wherein the branch outcome queue is a FIFO buffer and
2 data is transmitted to and from the buffer using an error correction technique.

1 15. The SRT processor of claim 12 wherein the individual outcomes stored in the branch
2 outcome queue comprise:

04838078-041501
FO5TH0-B/08E350

3 a program type classifying the branch instruction; and
4 a target address corresponding to the instruction to be executed immediately following the
5 branch instruction;
6 wherein during execution of the second program thread, the SRT processor may identify
7 the appropriate branch instruction using the program counter value and may also fetch instructions
8 ahead of the branch instruction using the target address.

1 16. The SRT processor of claim 12 wherein if the branch outcome queue becomes full, the first
2 thread is stalled to prevent more branch outcomes from entering the branch outcome queue; and
3 wherein if the branch outcome queue becomes empty, the second thread is stalled to allow
4 more branch outcomes to enter the branch outcome queue.

1 17. A method of predicting branch instructions in an SRT processor which can fetch and
2 execute a set of instructions in two separate threads so that each thread includes substantially the
3 same instructions as the other thread, one of said threads being a leading thread and the other of
4 said threads being a trailing thread, the method comprising:

5 training a branch predictor to store predicted outcomes from branch instructions in the
6 leading thread;

7 probing the branch predictor to predict outcomes of future executions of branch instructions
8 in the leading thread;

9 storing actual outcomes of branch instructions in the leading thread in a branch outcome
10 queue;

11 probing the branch outcome queue to predict outcomes of corresponding branch
12 instructions in the trailing thread.

1 18. The method of claim 17 further comprising:
2 executing the branch instructions in the leading and trailing threads in program order.

1 19. The method of claim 18 further comprising:
2 storing the actual outcomes of branch instructions in the leading thread in the branch
3 outcome queue after the branch instructions retire;
4 wherein the outcomes are identified by a branch identifier and a target address signifying
5 the subsequent instruction to be executed as a result of the outcome of the execution of the branch
6 instruction.

1 20. The method of claim 18 further comprising:
2 using a FIFO buffer as the branch outcome queue;
3 wherein if the buffer becomes full, the leading thread is stalled to prevent more branch
4 outcomes from entering the buffer; and
5 wherein if the buffer becomes empty, the trailing thread is stalled to allow more branch
6 outcomes to enter the buffer.

1 21. The method of claim 18 further comprising:
2 transmitting data to and from the branch outcome queue using an error correction
3 technique.